**Requirements**

**Virtual Environment**

**Virtual Environment Commands**

| **Command** | **Linux/Mac** | **Windows/GitBash** |
| --- | --- | --- |
| Create | python3 -m venv venv | python -m venv venv |
| Activate | source venv/bin/activate | source venv/Scripts/activate |
| Install | pip install -r requirements.txt | pip install -r requirements.txt |
| Deactivate | deactivate | deactivate |

certifi==2025.6.15

charset-normalizer==3.4.2

idna==3.10

requests==2.32.4

urllib3==2.5.0

VSCode

import pandas as pd

import sqlite3

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from wordcloud import WordCloud

**Installing SQLite on Windows**

For Windows machines:

1. Download the **sqlite-tools-win-x64-3490100.zip** file and unzip it.
2. From your [git-bash terminal](https://www.codecademy.com/article/command-line-setup), open the directory of the unzipped folder with cd ~/Downloads/sqlite-tools-win-x64-3490100.zip/sqlite-tools-win-x64-3490100.zip/. 3.Try running sqlite with the command winpty ./sqlite3.exe. If that command opens a sqlite> prompt, congratulations! You’ve installed SQLite.

We want to be able to access this command quickly from elsewhere, so we’re going to create an alias to the command. Exit the sqlite> prompt by typing in Ctrl + C, and in the same git-bash terminal without changing folders, run these commands:

echo "alias sqlite3=\"winpty ${PWD}/sqlite3.exe\"" >> ~/.bashrc

and

source ~/.bashrc

the **ipykernel** package, which is needed for running Python cells in Jupyter, is missing from your current Python environment. To fix this, follow these steps:

1. **Open your Command Prompt or Terminal:**Ensure you're using the same Python interpreter that your Jupyter Notebook is configured to use (in your case, Python 3.13.2 located at c:/Python313/python.exe).
2. **Run the Installation Command:**Copy and paste the following command to install (or reinstall) the ipykernel package: c:/Python313/python.exe -m pip install ipykernel -U --user --force-reinstall

This command will:

* Use the specified Python interpreter.
* Install or update ipykernel.
* Ensure it's installed in the user space (if you don’t have administrative rights).
* Force a reinstallation even if it’s already present.

1. **Restart Jupyter Notebook:**After the installation completes, restart your Jupyter Notebook to let it recognize the new package.
2. **Verify the Kernel:**Once restarted, ensure that the notebook is using the correct kernel. If necessary, select the Python 3.13.2 kernel from the notebook’s kernel menu.

The Bash command line, also known as the terminal or shell, is a text-based interface used to interact with the operating system. It allows users to execute commands, manage files, and run programs. Here are some fundamental aspects of using the Bash command line:

**Basic Navigation and File Management**

* pwd: Print working directory, displays the current directory path.
* ls: List directory contents, shows files and subdirectories in the current directory.
  + ls -l: List in long format, displaying detailed information.
  + ls -a: List all, including hidden files and directories.
* cd: Change directory, navigates to a specified directory.
  + cd ..: Move one level up.
  + cd ~: Go to the home directory.
* mkdir: Make directory, creates a new directory.
* rmdir: Remove directory, deletes an empty directory.
* rm: Remove, deletes files or directories.
  + rm -r: Remove recursively, deletes a directory and its contents.
* cp: Copy, duplicates files or directories.
* mv: Move, relocates or renames files or directories.
* touch: Create an empty file or update the timestamp of an existing file.

**Working with Files**

* cat: Concatenate and display, shows the content of a file.
* less or more: View file content page by page.
* head: Display the first few lines of a file.
* tail: Display the last few lines of a file.
* grep: Search for a pattern in files.
* echo: Print text to the terminal.

**Permissions**

* chmod: Change permissions, modifies file access rights.

Process Management

ps: Display running processes and kill: Terminate a process.

Other Useful Commands

history: Show command history, clear: Clear the terminal screen, man: Display the manual page for a command, sudo: Execute a command with superuser privileges, and df: Display disk space usage.

Example Usage

|  |
| --- |
| # Navigate to the home directory cd ~  # Create a new directory named "mydir" mkdir mydir  # Change into the new directory cd mydir  # Create a file named "myfile.txt" touch myfile.txt  # List all files in the current directory ls -l  # Go back to the parent directory cd ..  # Remove the directory "mydir" and its contents rm -r mydir |

def module\_func():

    print("you just imported the user-defined module")

import my\_module

my\_module.module\_func()

**TECHNOLOGIES USED:**

**Languages & Core Libraries:**  
*Python* – primary language for data analysis and project development  
*Pandas* – for data cleaning, transformation, and analysis

**Visualization Tools:**  
*Matplotlib* – for basic charts and static visualizations  
*Plotly* – for interactive graphs and visual exploration  
*Tableau* – for building interactive dashboards and communicating insights visually

**Data Storage & Querying:**  
*SQLite* – for lightweight, file-based data storage and SQL querying

**Development Environment:**  
*Jupyter Notebook* – for exploratory data analysis and documenting workflows  
*VS Code* – for script development and file management

**Version Control & Documentation:**  
*Git & GitHub* – for version control and project sharing  
*Markdown* – for notes, documentation, and explanations

**Data Sources:**

[https://www.kaggle.com/datasets/kaggleprollc/air-quality-data-nyc\](https://www.kaggle.com/datasets/kaggleprollc/air-quality-data-nyc%5C) <https://www.kaggle.com/datasets/nycparks/tree-census>